

Tabla de Contenidos



- [Java List default Method removeif](#)
- [Java List default Method replaceAll](#)
- [Java List y conclusiones](#)
- [Otros artículos relacionados](#)

El uso de Java List incluye mejoras a partir de Java 8 y muchas veces nos olvidamos de ellas . Vamos a hablar un poco del interface List en Java 8 y como nos puede ayudar en nuestro día a día. Estamos acostumbrados a recorrer una lista con un bucle foreach.

```
import java.util.List;

public class Principal {

    public static void main(String[] args) {
        List<String> lista= List.of("hola",
"que","tal","estas");
        for (String texto: lista) {
            System.out.println(texto);
        }

    }

}
```

El resultado simplemente nos imprime los datos en la consola:

```
hola
que
tal
estas
```

Podemos usar las capacidades de Java 8 y convertir la lista en un Stream para recorrerla de una forma mucho más compacta:

```
import java.util.List;

public class Principal2 {

    public static void main(String[] args) {
        List<String> lista= List.of("hola",
"que", "tal", "estas");
        lista.stream().forEach(System.out::println);
    }

}
```

El código es mas compacto y una función forEach que recibe un Consumer para procesar la lista de forma compacta. El resultado es idéntico acabamos de usar programación funcional y es un avance importante.

```
hola
que
tal
estas
```

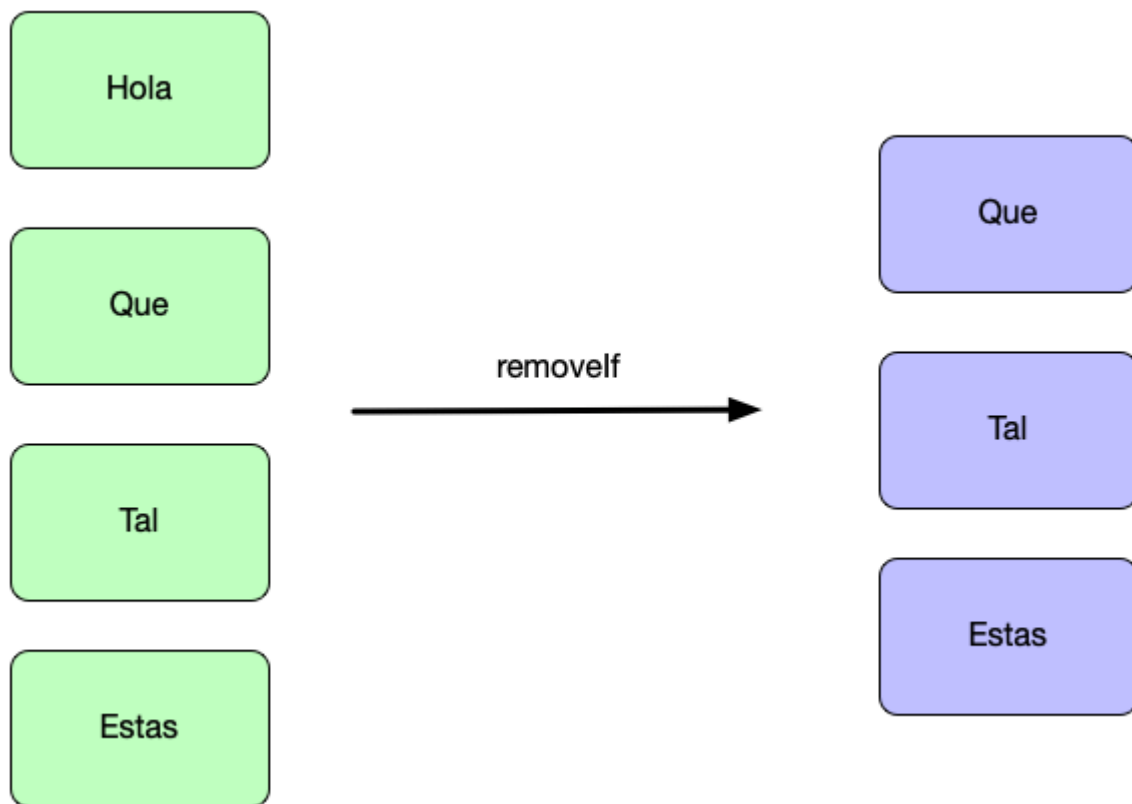
Java List default Method removeIf

El interface List soporta varios default Methods que nos pueden ser muy útiles en nuestro día a día. Uno es removeIf que elimina elementos dependiendo de una condición veámoslo en acción:

```
import java.util.ArrayList;
import java.util.List;
```

```
public class Principal2 {  
  
    public static void main(String[] args) {  
        List<String> lista= new ArrayList<>();  
        lista.add("hola");  
        lista.add("que");  
        lista.add("tal");  
        lista.add("estas");  
        lista.stream().forEach(System.out::println);  
        lista.removeIf((e)->e.equals("hola"));  
        System.out.println("*****");  
        lista.stream().forEach(System.out::println);  
    }  
  
}
```

El resultado es muy directo elimina los elementos que cumplan con la condición:



```
hola  
que  
tal  
estas
```

```
*****
```

```
que  
tal  
estas
```

Recordemos que para eliminar elementos tenemos que haber creado una lista “mutable” previamente. El método `List.of` construye una lista inmutable.

Java List default Method `replaceAll`

De igual manera que podemos eliminar elementos recorriendo la lista podemos reemplazarlos todos de golpe usando también programación funcional y apoyándonos en el

método replaceAll

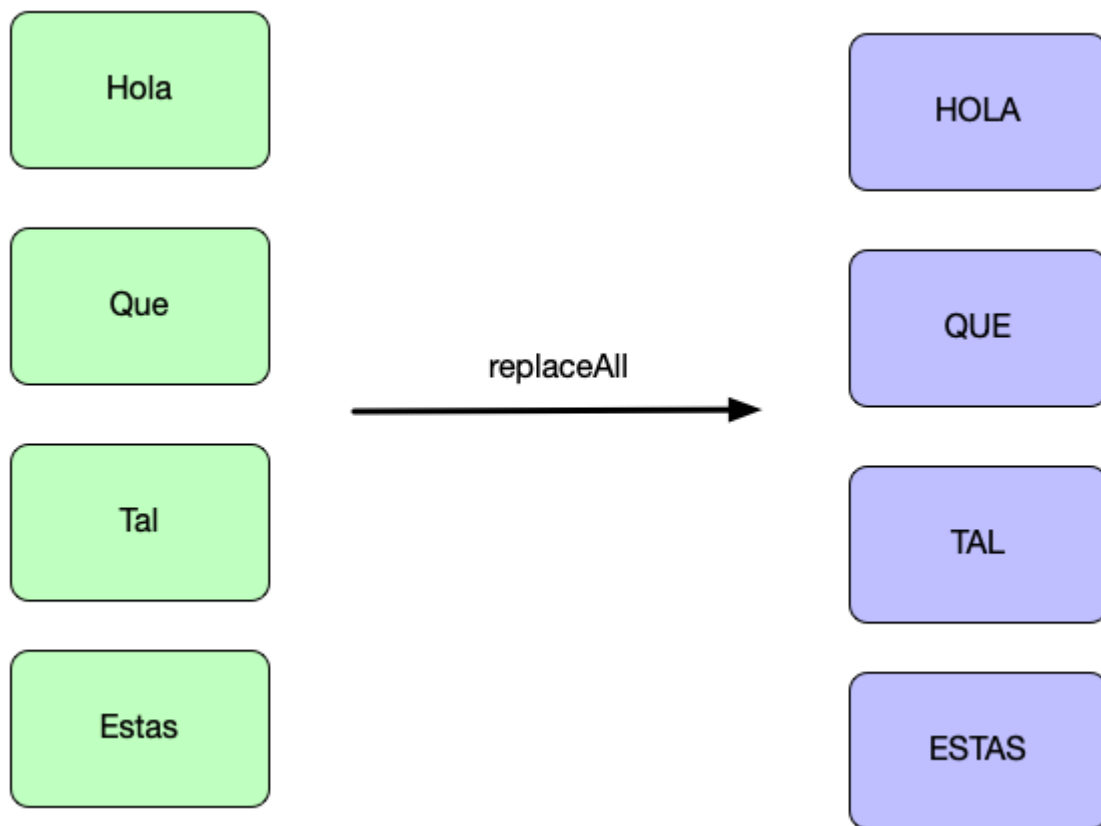
```
import java.util.ArrayList;
import java.util.List;

public class Principal3 {

    public static void main(String[] args) {
        List<String> lista= new ArrayList<>();
        lista.add("hola");
        lista.add("que");
        lista.add("tal");
        lista.add("estas");
        lista.stream().forEach(System.out::println);
        lista.replaceAll((e)->e.toUpperCase());
        System.out.println("*****");
        lista.stream().forEach(System.out::println);
    }

}
```

Nos reemplaza todos los elementos:



El resultado lo podemos ver:

```
hola
que
tal
estas
*****
HOLA
QUE
TAL
ESTAS
```

Java List y conclusiones

A veces se nos olvida que todas las APIs van evolucionando y el interface List no es una excepción. Recordemos que tenemos poco a poco que acostumbrarnos a la programación

funcional para poder sacar mayor partido a nuestras clases.

Otros artículos relacionados

- [Java String to Date utilizando Java 8](#)
- [File to String Java 8 y manejo de ficheros](#)
- [Java 8 Functional Interfaces y sus tipos](#)
- [Java Stream String y Java 8](#)
- [ArrayList](#)